Año 3 nº 1, abril 2019.
Propietario y editor:
Asociación de Docentes de Música.
Queda hecho el depósito que dispone la
Ley 11.723. Exp. N° 5343007.
ISSN 2545-7802.
www.adomu.com.ar pp. 65-71

Boletín de Investigación Educativo-Musical / Asociación de Docentes de Música
Retomando la labor pionera del *Collegium Musicum de Buenos Aires* (1993-2009)

CONFERENCIA

UN ENTORNO EDUCATIVO DE PROGRAMACIÓN VISUAL DESTINADO A LA SÍNTESIS Y AL PROCESAMIENTO DE SONIDO Y MÚSICA EN TIEMPO REAL

Pablo Cetta
Universidad Católica Argentina
pablo.cetta@uca.edu.ar

1. Introducción

El propósito del entorno *WAUL (Web Audio Library)* es la utilización facilitada de la *Web Audio API*[1] a través de una interfaz gráfica que permite la programación empleando objetos interconectables por medio de cables virtuales. El desarrollo de la interfaz fue realizado partiendo de la librería *p5.js*[2] , creada por Lauren McCarthy, y en particular *p5.dom*, que permite interactuar con distintos elementos presentes en *HTML* 5. *WAUL* fue creado en principio con fines didácticos, dado que la posibilidad de ser utilizado desde un navegador de Internet –sin necesidad de instalar aplicación alguna- hace propicia la tarea de difundir y practicar diversos procesos de tratamiento digital de señales de audio, utilizables en música, desde un sitio especialmente creado o una plataforma de *e-learning*.

[1] *Web Audio* es una API destinada al procesamiento y síntesis de audio digital en aplicaciones web, desarrollada por el *Audio Working Group* del *W3C Consortium.* La documentación más reciente puede consultarse en https://webaudio.github.io/web-audio-api/
[2] https://p5js.org/

La *Web Audio API* ya se encuentra implementada en gran parte en las versiones más recientes de los navegadores actuales. Si bien se la considera aún un diseño experimental, su utilización permite ver la enorme potencialidad de este desarrollo. La librería *WAUL* no cubre todas sus características aún, si bien agrega otras, gracias a la posibilidad de extender la API a través de la programación en *JavaScript*.

2. Programación con WAUL

La lógica del flujo de datos utilizada por esta librería intenta seguir los criterios empleados en otros lenguajes de procesamiento en tiempo real, como *Pure Data*[3] o *Max-MSP*[4] . Incluso los nombres de los objetos son en muchos casos iguales o similares.

La programación se realiza sobre una ventana -un *canvas*-, en cuya parte superior se destaca un menú desde el cual se puede iniciar el procesamiento de audio, crear nuevos *patches* o recurrir a otros preexistentes, almacenados en el servidor o bien de forma local en el ordenador del usuario.

Los objetos que conforman la librería se dividen en dos grupos: objetos de control –aquellos que responden a un mensaje particular, en general producido por el usuario- y los objetos de audio – objetos que producen datos a la frecuencia de muestreo, cada vez que es iniciado el procesamiento digital de señales. Al igual que en los lenguajes antes citados, la lógica de programación se basa en un *hot inlet* –el primero de la izquierda, que dispara las operaciones- y *cold inlets* –aquellos que almacenan los datos, hasta el momento en que se recibe un valor a través del *hot inlet*.

Entre los dispositivos de control disponemos de objetos que realizan operaciones matemáticas, trigonométricas, operadores relacionales, operadores de conversión y objetos de control temporal (gatillos, retardos y metrónomo). Además, contamos con objetos de interfaz con el usuario, tales como editores para mensajes alfanuméricos, botones, *sliders* y generadores gráficos de envolventes.

[3] https://puredata.info/
[4] https://cycling74.com/

Como es propio de los lenguajes basados en una interfaz gráfica, la posibilidad de contar con conexiones remotas facilita la comunicación entre objetos y abstracciones.

Por otra parte, entre los objetos de audio disponibles hallamos los clásicos conversores *adc~* y *dac~*, generadores de señales –osciladores con distintas señales periódicas y ruido blanco-, dispositivos de reproducción y grabación de archivos de audio, líneas de retardo, filtros, generadores de envolventes, convolución aplicable a la síntesis cruzada y a la reverberación natural, compresión dinámica, análisis espectral y espacialización.

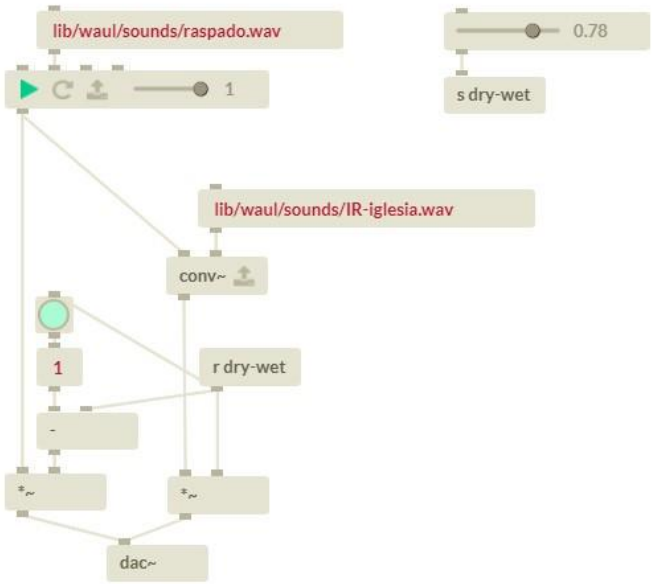

Figura 1. Reverberación natural empleando convolución

El uso del conversor analógico-digital se halla reservado para los casos en los que la librería se encuentra instalada en un servidor seguro, pues de otro modo el mismo navegador impide el acceso a los dispositivos de entrada, por cuestiones de seguridad.

La reproducción de archivos de audio se realiza a través de dos técnicas: mediante la lectura directa del archivo, o a través de un *buffer*. En el segundo caso, es posible variar la velocidad de lectura, y en ambos casos recurrir a archivos locales o a archivos alojados

en el servidor, y la reproducción repetitiva. Los filtros disponibles, por otra parte, son de un polo, de un cero – ambos con la posibilidad de aumentar la cantidad de muestras de retardo, para crear así otras configuraciones- y el filtro bicuadrático.

El control de los distintos parámetros de los objetos de audio se realiza a través de constantes, o bien mediante envolventes, tanto numéricas como gráficas.

La *Web Audio API* propone configuraciones interesantes en cuanto al tratamiento multicanal. La librería *WAUL* implementa, por el momento, la posibilidad de procesar señales mono o estereofónicas. Lo mismo ocurre con las posibilidades de espacialización sonora. Si bien contamos en *WAUL* solamente con control de lateralización por *intensity panning*, se trabaja actualmente en la incorporación de síntesis binaural y otras técnicas multicanal.

En cuanto a las abstracciones –*patches* almacenados en disco que pueden ser incluidos en el *patch* principal- la librería admite un solo nivel de encapsulamiento. La comunicación entre el programa principal y los subprogramas se realiza exclusivamente mediante conexiones remotas, locales y globales.

Figura 2. Uso de abstracciones. Síntesis aditiva aleatoria

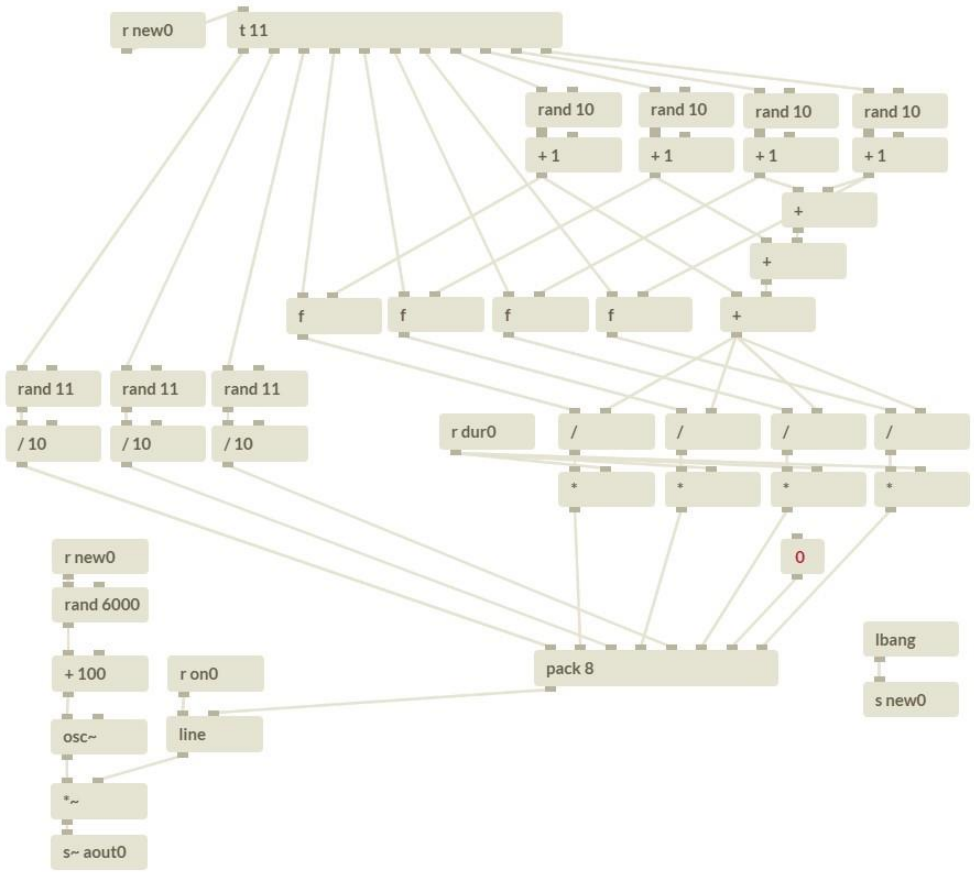

Figura 3. Abstracción (objeto *rosc*) de ejemplo de síntesis aleatoria de la Figura 2

3. Descarga e instalación

A fin de probar la librería y proceder a la descarga nos dirigimos a:

https://www.pablocetta.com/waul.php

El archivo comprimido descargable contiene un sitio web básico preconfigurado, listo para copiar en un servidor. La instalación incluye la librería y sus dependencias, archivos *json* de configuración, archivos *css* de estilos, los *patches* comentados en el tutorial y archivos de audio, entre los que se encuentran varias respuestas a impulso para emplear en convolución.

En la página de descargas encontraremos también la documentación de la librería, con *patches* de ejemplo, combinaciones de teclas necesarias para la programación y una guía de referencia de los objetos.

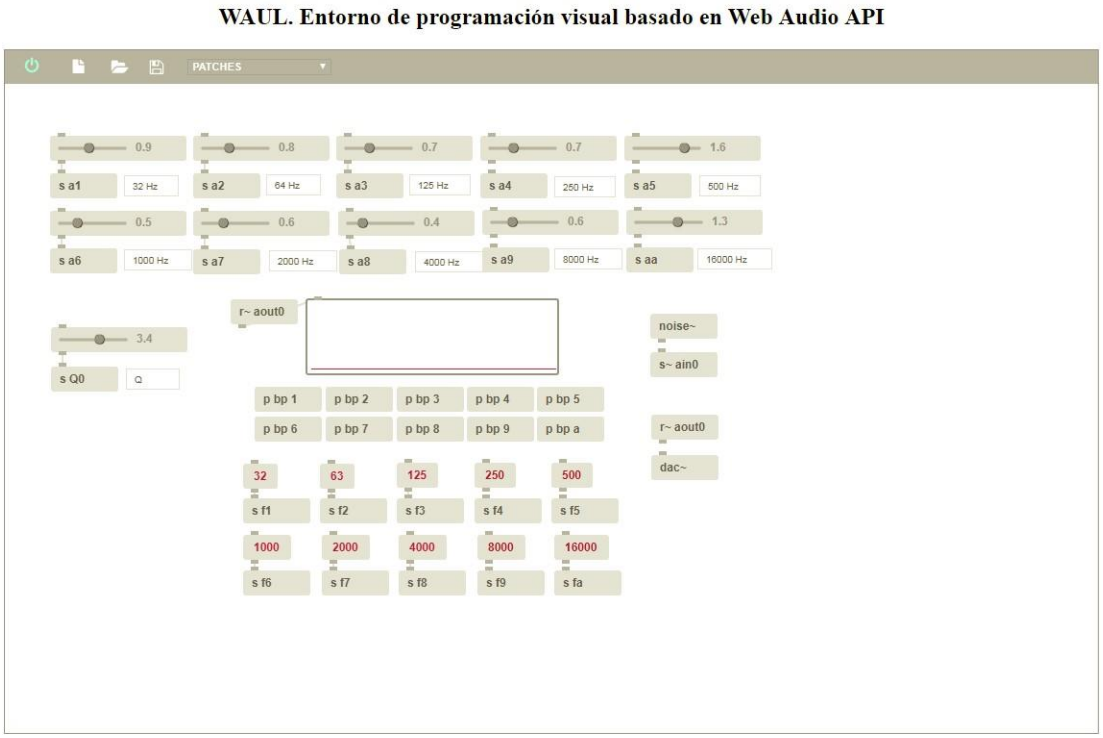

Figura 4. Página de inicio del sitio preconfigurado

Se recomienda la utilización de *WAUL* en versiones actuales del navegador *Google Chrome*, dado que aún no ha sido probado en profundidad en otros navegadores.

Si bien un sitio puede contener varios entornos de programación en *WAUL*, *Web Audio API* sólo admite un entorno por cada página web. Dicho de otro modo, si deseo contar con dos o más aplicaciones en un mismo sitio, cada una de ellas debe residir en una página -archivo *html* o *php*- diferente.

4. Conclusiones

Esta versión preliminar de la librería *WAUL* propone optimizar determinados usos de la *Web Audio API*, en particular, aquellos orientados a la programación de aplicaciones que

permitan la comprensión de procesos de audio digital. El uso de una interfaz gráfica no sólo simplifica la generación de programas por parte de usuarios no expertos en el uso de lenguajes simbólicos, sino que además contribuye al seguimiento y comprensión del flujo de la información contenida en ellos. A partir de los alcances logrados, cabe esperar el agregado de nuevas prestaciones y ampliaciones, que potencien las capacidades del desarrollo aquí presentado.

Referencias bibliográficas

Cetta, P. (2014). *Captura y procesamiento de sonido*. Buenos Aires: Universidad Nacional de Quilmes.

McCarthy, L. y otros. *P5.js Reference*. Consultado en: https://p5js.org/

Smus, B. (2013). *Web Audio API*. Sebastopol, CA: O´Reilly.

WebAudio API W3C Editor´s Draft. January 2018. Consultado en: https://webaudio.github.io/web-audioapi

Pablo Cetta

Doctor en Música (UCA). Compositor. Becario de la Fundación Antorchas, del L.I.P.M., de la Fundación Rockefeller (intercambio con las universidades de California en San Diego y Stanford), del Fondo Nacional de las Artes, del Institute International de Musique Electroacoustique de Bourges, del Laboratorio de Informática y Electrónica Musical de Madrid y del Ministerio de Educación y Cultura de España. Premios Municipal de Música, Argentores, Concurso Internacional de Bourges, Euphonies d'Or. Director del Instituto de Investigación Musicológica Carlos Vega (UCA) y del Doctorado en Música de la FACM (UCA).